DUE: On Blackboard, Friday, February 18 at 11:59PM. Upload your .java files as attachments and report who your team members were.

The goal of this lab is figure out the order in which methods call one another and in which the calls return.

New thing: Form teams of three to four people for this lab!

USE A SEPARATE DIRECTORY for Lab 03! Each team member will use his or her own “S:” drive and directory however.

To be done in the lab. Complete at home if necessary. Book Classes and media resources will not be used.

To begin, form your teams, and everybody make your new directory and download into it the three .java files linked from Week 4 of http://www.cs.albany.edu/~sdc/CSI310. The lab assignment has these 4 phases:

Phase 0 The team together figures out on paper what the 3 class program below prints.

Then the team members get the program into their directories, compile and run it, and see whether they were right.

Phase 1 The team together figures out on paper what the 3 class program below prints when phaseNumber has been initialized to 1.

Then the team members modify the program in their directories, compile and run it, and see whether they were right.

Phase 1X phaseNumber remains initialized to 1.

Each team member adds additional method calls in an interesting way according to the instructions in the code. The new code must compile and run before moving on. Each team member will turn in (now or later, for credit) all three .java files including those that he or she has modified.

The team members switch seats. Each team member looks at another member’s modified program. Each team member writes ON PAPER a prediction of what it will print when run. Running the program before writing the prediction is not fair.

Then, the team reviews all the predictions, runs all the programs, and sees how accurate are the predictions.

Phase 2 Together, the team predicts what the program prints when phaseNumber is initialized to 2.

Then when they think they have figured it out, they run the program to check.

/**
 * Do not modify except for changing the phase number
 * from 0 to 1 and then later to 2.
 */
class Application
{
  public static void main(String[] a)
  {
    int phaseNumber = 0;
    Dog rmainsDog = new Dog();
    if(phaseNumber==0)
    {
      System.out.println("main calling doOne on Dog " + rmainsDog);
      rmainsDog.doOne();
      System.out.println("main calling doTwo on Dog " + rmainsDog);
      rmainsDog.doTwo();
    }
Cat rmainsCat = new Cat(rmainsDog);
if(phaseNumber==1)
{
    System.out.println("main calling catActOne on Cat " + rmainsCat);
    rmainsCat.catActOne();
}
if(phaseNumber==2)
{
    System.out.println("main calling countDown on Cat " + rmainsCat);
    rmainsCat.countDown(5);
}
System.out.println(" main will now return. Goodbye.");
}

class Dog
{
    /**
     * The Dog's constructor. Do not modify.
     */
    public Dog()
    {
        System.out.println("Dogs constructor called with this=" + this);
        System.out.println(" Dogs constructor with this=" + this + " returning.");
    }
    /**
     * MODIFY this method by adding calls to the doTwo method.
     */
    public void doOne()
    {
        System.out.println("Dogs doOne method called with this=" + this);
        //PUT IN YOUR OWN METHOD CALLS HERE.
        System.out.println(" Dogs doOne method with this=" + this + " returning.");
        return;
    }
    public void doTwo()
    {
        System.out.println("Dogs doTwo method called with this=" + this);
        //Put additional method calls here AT YOUR OWN RISK!
        System.out.println(" Dogs doTwo method with this=" + this + " returning.");
        return;
    }
}
class Cat {
    public Dog pDogIKnow;
    /**
     * Cat's constructor. Do not modify.
     */
    public Cat(Dog x) {
        System.out.println("Cats constructor called with this=" + this + " and Dog x=" + x);
        pDogIKnow = x;
        System.out.println(" Cats constructor with this=" + this + " returning.");
        return;
    }
    /**
     * Please add additional calls to methods of the
     * Dog which this Cat knows.
     */
    public void catActOne() {
        System.out.println("Cats catActOne method called with this=" + this);
        //PUT IN YOUR OWN METHOD CALLS HERE, like
        pDogIKnow.doOne();
        System.out.println(" Cats catActOne method with this=" + this + " returning.");
        return;
    }
    /**
     * Do not modify this method.
     */
    public void countDown(int i) {
        System.out.println("countDown method called with i=" + i + " and this=" + this);
        if(i < 0) {
            System.out.println("Cats HATE NEGATIVE NUMBERS.MMEEEOOOWWW.");
        } else if(i == 0) {
            System.out.println("countDown with i=" + i + " and this=" + this + " is DONE COUNTING");
        } else {
            System.out.println("countDown with i=" + i + " and this=" + this + " RECURSING NOW.");
            this.countDown( i - 1 );
            System.out.println("RECURSION returned to where i=" + i);
        }
        System.out.println(" countDown method with i=" + i + "and this=" + this + " returning.");
        return;
    }
}